Facultad de Ciencias

Predicción con datos SCADA de series temporales

(Time series forecasting for SCADA data)

Trabajo de Fin de Máster

para acceder al

MÁSTER EN CIENCIA DE DATOS

Autor: Miguel Martínez-Conde Salamanca
Director: Sixto Herrera García
Co-Director: Carlos A. Meneses Agudo
10 de junio de 2020

Índice general

Resumen

El análisis de series temporales ha adquirido un gran protagonismo en el ámbito académico dadas sus implicaciones en áreas como la ingeniería, las finanzas o las ciencias sociales. El objetivo en este caso no es ahondar en dicho ámbito, sino trasladarlo a un entorno industrial con series temporales que presentan características más dispares y, en ocasiones, intermitencias.

En la presente memoria se realiza una implementación de varios modelos de machine learning acerca de datos SCADA (Supervisory Control And Data Acquisition) de series temporales referidas a consumos de energía en una planta industrial con el objetivo de evaluar sus resultados para la toma de decisiones operativas en este contexto. Los modelos que se implementarán, previa aplicación de técnicas de curación y preprocesado de datos, serán máquinas de vectores soporte, modelos aditivos generalizados y modelos de ensembles en sus vertientes de bagging y boosting. Un modelo final será propuesto como una combinación de todos los anteriores adoptando de nuevo una metodología de ensembles y los modelos stacking, mediante la estimación de un meta-modelo.

En última instancia, se elaborarán las predicciones para diferentes horizontes temporales que permitan obtener conclusiones sobre la viabilidad de la aplicación de estas técnicas en un ámbito centrado en la toma de decisiones operativas y que contribuya de forma aditiva a un análisis similar como el elaborado por Meneses Agudo et al. (2019).

Palabras clave: Series temporales, SCADA, machine learning, predicción.

Abstract

Time series analysis has gained weight in the academic literature because of its growing penetration in areas such as engineering, finance or social sciences. The objective of the present thesis is not to research further and get a deeper look at these techniques, but rather translate their application to the industrial sector, in time series with heterogenous characteristics and, in some occasions, and intemittent behaviour.

In this thesis we implement several machine learning models to time series of SCADA data (Supervisory Control and Data Acquisition) referring to energy consumption in an industrial plant, with the objective to evaluate the appropiateness of their results in order to improve the firm's decision making. Previous to the application of these models, curation and preprocessing techniques will be applied to the data. The specific models applied are Support Vector Machines, Generalized Additive Models and Ensembles models in their bagging and boosting variations. A final model will be propposed as a combination of the mentioned models through a new ensembles methodoloy and stacking models, with the estimation of a meta-model.

As a final note, we will forecast the objective variables in different time horizons so as to analyse the capability of these techniques to improve decision making, and complementing previous research as shown by Meneses Agudo et al. (2019).

Keywords: Time series, SCADA, machine learning, forecasting.

Acrónimos

- **UC** = Universidad de Cantabria
- **CIC** = Consulting Informático Cantabria
- **SCADA** = Supervisory Control and Data Acquisition

Conjunto de datos

- **Pot.** = Agua potable (m^3)
- **Des.** = Agua desionizada (m^3)
- **Perm.** = Agua permeada (m^3)
- **GDir.** = Gas Directo (kWh)
- **Tec.** = Gas Tecnológico (kWh)
- **Cal.** = Gas calefacción (kWh)
- **Cli.** = Climatizadores (kWh)
- **Fr.** = Frío (kWh)
- **Al.** = Alumbrado (kWh)
- **Ins.** = Instalaciones (kWh)
- **ACom.** = Aire Comprimido (kWh)
- **Prod1.** = Línea de producción 1 (N^o automóviles)
- **Prod2.** = Línea de producción 2 (N^o automóviles)
- **Prod3.** = Línea de producción 3 (N^o automóviles)
- **Tint.** = Temperatura en la instalación ($^\circ C$)
- **Text.** = Temperatura exterior ($^\circ C$)

Métodos

- **GAM** = Generalized Additive Model
- **SVM** = Support Vector Machine
- **GBDT** = Gradient Boosting Decision Trees
- **RF** = Random Forest
- **LightGBM** = Light Gradient Boosting Machine
- **XGBoost** = Extra Gradient Boosting
- **Catboost** = Categorical Boosting
- **ARIMA** = Autoregressive Integrated Moving Average
- **VAR** = Autoregressive Vector
- **SARIMA** = Seasonal Autoregressive Integrated Moving Average

CAPÍTULO 1

Introducción

La energía eléctrica juega un papel fundamental en el desarrollo económico y social en el mundo actual. Por esto, la predicción precisa de las cargas de consumo eléctrico empleadas tanto por los productores como por los consumidores de energía es básica para asegurar los suministros y reducir costes de almacenamiento (He, 2017), adquiriendo especial relevancia en el ámbito industrial. En este contexto, las cargas de energía eléctrica suponen un campo interesante de aplicación de técnicas de aprendizaje automático a series temporales [1] debido a la disponibilidad de grandes conjuntos de datos con suficientes patrones tanto lineales como no lineales (Robinzonov et al., 2012).

Concretamente, la utilización de estas herramientas de predicción es de gran utilidad en el ámbito industrial con el fin de planificar estrategias de consumo que minimicen sus costes y, por tanto, maximicen sus beneficios. En este caso, se utilizarán modelos que se engloban en lo que en la literatura se denomina *"Load Forecasting Techniques"* o de predicción de carga eléctrica, añadiendo otros que se encuadran dentro del aprendizaje máquina.

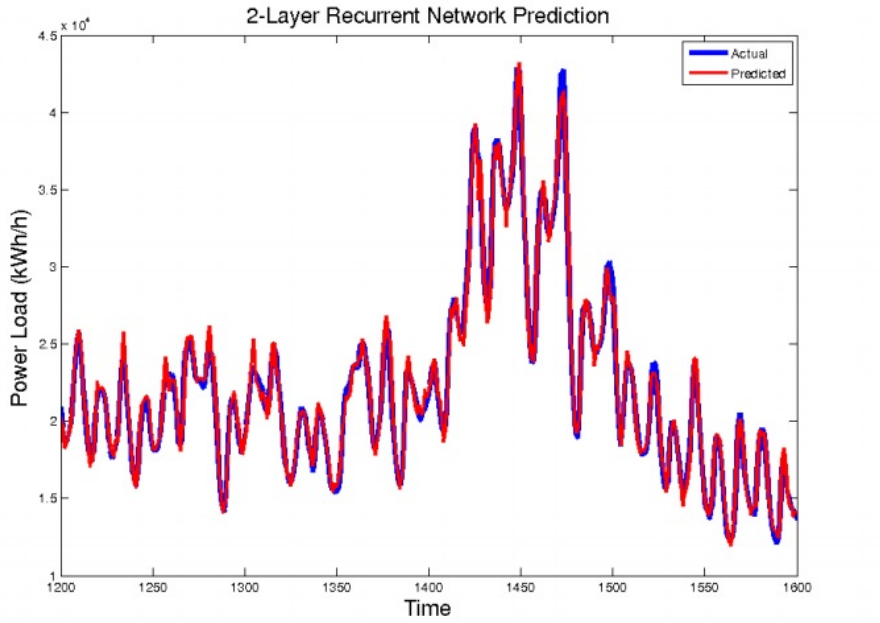
Este es un campo muy extenso y explotado en los últimos años, que cuenta con una batería muy amplia de modelos. Los más empleados son los basados en redes neuronales, ya sea de forma individualizada (Damborg et al., 1990) o bien componiendo modelos híbridos (Juberias et al., 1999), a los cuales se añaden los modelos clásicos (Elrazaz and Mazi, 1989) de predicción de series temporales.

En la batería de modelos que componen la literatura de predicción de carga eléctrica (Singh et al., 2012) destacan las técnicas tradicionales (Regresión lineal, Alisado Exponencial...), tradicionales modificadas (ARIMA, SARIMA, SVMs...) y de computación suave o *"Soft Computing Techniques"* (Algoritmos Genéticos, *"Fuzzy Logic"*, Redes Neuronales...). Los métodos más frecuentes en esta literatura son las redes neuronales y las SVMs, como se ilustra en los ejemplos siguientes.

[1] **Serie Temporal**: registro de cualquier cantidad fluctuante medida de forma ordenada en diferentes instantes de tiempo (Abril, 2011)

El trabajo de Busseti et al. (2012) implementa varios modelos de *"Deep Learning"* con redes neuronales tanto de tipo *"Feedforward"* como Recurrentes, aplicadas a la predicción de carga eléctrica basándose únicamente en datos horarios y de temperatura para la *"Global Energy Forecasting Competition 2012 - Load Forecasting"*. Aplicando una red neuronal recurrente de dos capas, los resultados de predicción son los siguientes:

Figura 1.1: Predicción de carga eléctrica con redes neuronales recurrentes

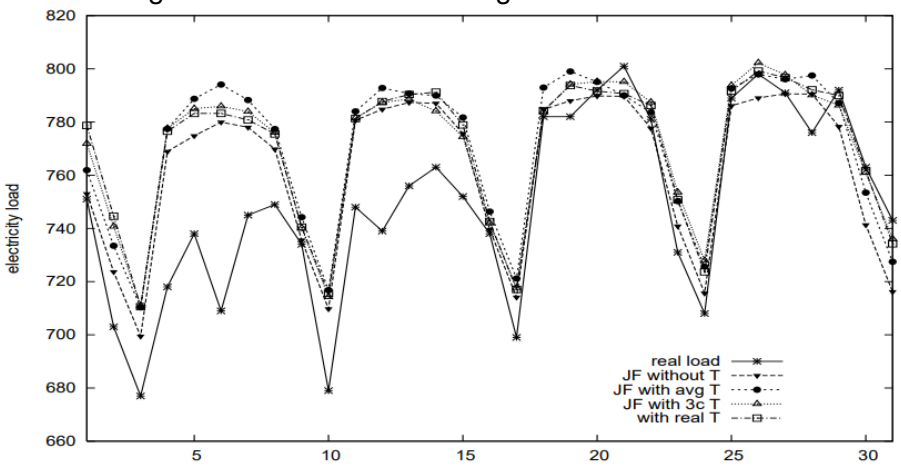


Fuente: Busseti et al. (2012)

En la Figure 1.1 se puede observar una precisión relativamente alta en la predicción del fenómeno.

Por otro lado, Chen et al. (2004) elaboran un análisis similar. En él, se sirve de datos de demanda de carga eléctrica entre los años 1997 y 1999 para elaborar predicciones con SVMs en la *"EUNITE Competition 2001"*. Varios de sus resultados son los siguientes:

Figura 1.2: Predicción de carga eléctrica con SVMs



Fuente: Chen et al. (2004)

En la figura 1.2 se muestran los resultados de predicción de carga eléctrica con SVMs para una misma variable objetivo y teniendo en cuenta distintos regresores.

La presente memoria va a tratar de elaborar predicciones con calidad y escalabilidad suficientes como para ser útiles en un ámbito industrial. Así pues, se contará con un conjunto de datos de tipo SCADA[2] donde, además de tratar con este tipo de modelos de predicción de carga eléctrica, también van a añadirse un GAM (Modelo Aditivo Generalizado) y modelos de aprendizaje automático basados en ensembles [3]. Dentro de estos modelos, se introducirán las metodologías de GDBT (*"Gradient Boosting Decision Trees"*) con varias implementaciones, Bagging y un Stacking que combine la batería de modelos implementados, conformando un análisis complementario e incremental respecto al trabajo previo elaborado por Meneses Agudo et al. (2019).

La organización de la memoria comenzará con una descripción del proceso de preprocesado y curación de datos para, una vez obtenido el conjunto de datos final para la implementación de los modelos, se pasará a realizar una descripción metodológica de las distintas técnicas que se pondrán en práctica. En última instancia, se elaborarán predicciones tanto a corto como a medio plazo y se compararán los resultados de los diferentes métodos utilizados con los obtenidos por Meneses Agudo et al. (2019), de forma que se puedan obtener conclusiones fiables sobre la aplicación industrial de este tipo de modelado.

[2] **SCADA**: Supervisory Control and Data Acquisition.
[3] **Ensembles**: modelo construido en base a un conjunto de modelos individuales.

CAPÍTULO 2

Descripción y preprocesado de la base de datos

2.1. Descripción de la base de datos

Los datos que serán empleados a lo largo de la memoria provienen de una entidad perteneciente a la industria del automóvil a partir uno de sus sistemas SCADA (Supervisory Control and Data Acquisition). Dichos sistemas consisten en "*una aplicación o conjunto de aplicaciones de software especialmente diseñadas para funcionar sobre ordenadores de control de producción, con acceso a la planta mediante la comunicación digital con instrumentos y actuadores, e interfaz gráfica de alto nivel para el operador que permite la gestión y el control de cualquier sistema local o remoto"* (Pérez-López, Estaban 2015). Para la correcta gestión y control de la planta de producción dichos sistemas recogen datos de un gran número de parámetros dentro de la misma. En particular, en este estudio consideraremos el conjunto de variables reflejado en la Tabla 2.1.

Todas las variables fueron medidas con una frecuencia temporal de 15 minutos entre los años 2016 y 2018, dando lugar a un conjunto de datos compuesto de 15 variables, cada una con 101.472 observaciones.

2.2. Preprocesado y visualización

Para poder realizar el ajuste de los modelos de predicción descritos en el capítulo 3 ha sido necesaria la aplicación de varias técnicas de curación de datos, los cuales se describen a continuación.

2.2.1. Duplicados y valores ausentes

A la hora de proceder con el análisis y la visualización de las principales características de las series se obtuvieron errores asociados a valores duplicados en el conjunto de

Tabla 2.1: Descripción de variables (Ver Acrónimos)

Variable	Descipción	Medida
Agua potable	Agua con menor grado de salinidad, utilizada para actividades que requieren estas características de la misma (por ejemplo, es la empleada para procesos de pintura o algún otro que requiera de agua con menores grados de de cal ni sales).	m^3
Agua permeada	Agua con menor grado de salinidad, utilizada para actividades que requieren estas características de la misma (por ejemplo, es la empleada para procesos de pintura o algún otro que requiera de agua con menores grados de de cal ni sales).	m^3
Agua desionizada	Agua con menor grado de conductividad, utilizada para actividades que requieren estas características de la misma.	m^3
Gas Directo	Gas destinado al calentamiento de hornos y baños de cera.	kWh
Gas tecnológico	Referido al consumo de energía de agua sobrecalentada y convertida a gas.	kWh
Gas calefacción	Consumo energético para el mantenimiento de la temperatura de las instalaciones.	kWh
Climatizadores	Energía eléctrica consumida por los climatizadores de la instalación.	kWh
Frío	Energía eléctrica destinada a la producción del agua fría introducida en los climatizadores para el mantenimiento de la temperatura del taller.	kWh
Alumbrado	Energía eléctrica empleada en la iluminación interna de la instalación.	kWh
Instalaciones	Total de energía consumida en el taller, que engloba los valores de frío, alumbrado y climatizadores.	kWh
Aire comprimido	Variable encargada de la medición del consumo de kWh en los procesos de ensamblado y pintura.	kWh
Línea de producción 1	Cantidad de unidades producidas en la primera línea de producción.	N^o automóviles
Línea de producción 2	Cantidad de unidades producidas en la segunda línea de producción.	N^o automóviles
Línea de producción 3	Cantidad de unidades producidas en la tercera línea de producción.	N^o automóviles
Temperatura en la instalación	Temperatura dentro de la instalación.	$^\circ C$
Temperatura exterior	Temperatura en el exterior de la instalación.	$^\circ C$

Fuente: Elaboración propia

datos. Dichos duplicados se dieron en las fechas 30 de octubre de 2016, 29 de octubre de 2017 y 28 de octubre de 2018, todos ellos entre las 2:00 y las 2:45 horas por lo que se procedió a su eliminación.

Del mismo modo, se observó la existencia de observaciones ausentes en el dataset donde, en lugar de aparecer como NaN (Not a Number), simplemente no aparecían en el índice del conjunto de datos omitiendo su existencia. Para evitar problemas posteriores con dichos valores ausentes, éstos se introdujeron y se rellenaron con medias locales a través de una ventana deslizante centrada en la fecha ausente, considerando dos observaciones hacia atrás y otras dos hacia adelante.

Una vez corregidos los duplicados y los valores ausentes, se estimaron los porcentajes de NaNs y de valores anómalos para detectar problemas sistemáticos en alguna de las variables que limitasen su uso en el proceso de modelización. En este sentido, se concluyó no considerar las series de agua permeada y de agua desionizada debido al alto porcentaje de NaNs y de ceros, respectivamente, como se refleja en la Tabla 2.2.

Se puede comprobar como la variable referente al agua desionizada está prácticamente compuesta de ceros, con un 98,91 % mientras que la de agua permeada posee un 59,17 % de NaN, donde del 40 % restante el 38,22 % son ceros.

Para terminar, se llevaron a cabo dos acciones más:

Tabla 2.2: Porcentaje de NaN y ceros. Los valores destacados en negrita se corresponden con variables suprimidas por dicho valor.

	Pot.	**Des.**	**Perm.**	**GDir.**	**Tec.**	**Cal.**	**Cli.**	**Fr.**
NaN	0,85 %	0,85 %	**59,17 %**	0,85 %	0,00 %	0,75 %	29,20 %	0,00 %
Ceros	48,15 %	**98,91 %**	38,22 %	50,35 %	34,40 %	15,38 %	2,70 %	24,29 %
	Al.	**Ins.**	**AComp.**	**Prod 1**	**Prod 2**	**Prod 3**	**Tint**	**Text**
NaN	0,00 %	0,00 %	0,00 %	12,23 %	12,23 %	12,23 %	0,47 %	0,00 %
Ceros	8,40 %	0,00 %	42,90 %	40,70 %	48,78 %	45,41 %	0,00 %	0,00 %

Fuente: Elaboración propia

1. **Listwise**: El término listwise hace referencia a la eliminación por listas a la hora de manejar datos ausentes. Es decir, en el caso de que falte un valor determinado se procede a la eliminación del registro completo o, lo que es lo mismo, a la eliminación de todas las observaciones de las distintas variables en el instante del registro ausente. En este caso, la mayoría de las variables contenían valores ausentes únicamente en el comienzo y final de las series por lo que se ha aplicado esta técnica a dichas filas reduciendo ligeramente la muestra.

2. **Outliers**: Una vez homogeneizadas las series temporales de las diferentes variables se analizó la existencia de valores anómalos (outliers) los cuales pudieran estar asociados a errores en las mediciones. Si bien se identificaron algunos valores sospechosos, éstos correspondían a valores reales observados por lo que se decidió no realizar ningún tratamiento sobre ellos.

Por motivos de consistencia con el trabajo anterior de Meneses Agudo et al. (2019), se procedió a un cambio en la frecuencia temporal de las observaciones utilizando promedios por bloques de una hora. Esto tiene ciertos beneficios, como la ligera suavización de los valores extremos o la reducción del número de observaciones, que facilitará la implementación de los GAMs y las SVMs.

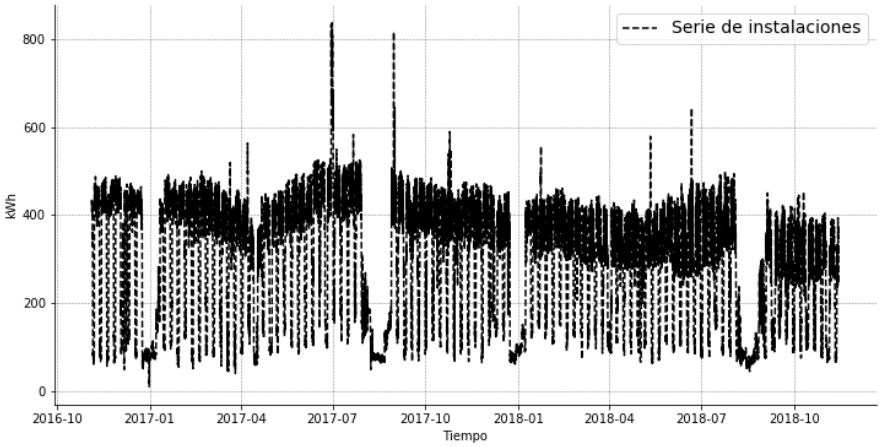
Una vez finalizados los pasos descritos anteriormente obtenemos una base de datos formada por 13 variables y 17.745 datos correspondientes al dato horario observado durante los años 2016, 2017 y 2018. A modo de ejemplo la Figura 2.1 muestra la serie final de la variable *Instalaciones*.

Notad que, dado que el conjunto de datos corresponden a una planta de producción de automóviles, con mediciones de la producción o del consumo en energía y calefacción, es esperable cierta correlación entre muchas de las variables. Por ello, inicialmente consideraremos todas las series, analizando posteriormente cuales entran o no en el modelo en base a las dependencias identificadas, así como las características y requerimientos de cada método considerado.

2.2.2. Escalado

Al considerar variables representando características diferentes del proceso de producción y, por tanto, con rangos y valores diferentes, la normalización de las series

Figura 2.1: Serie temporal de la variable Instalaciones.

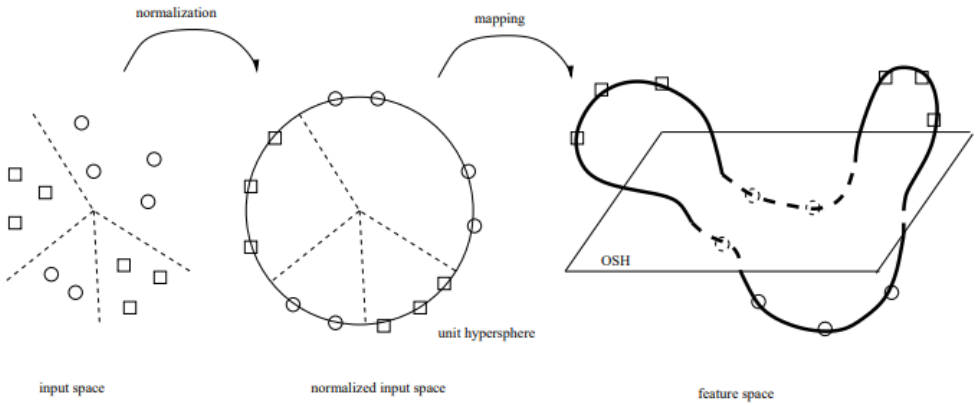


Fuente: Elaboración propia

temporales de cada variable, unificando su media (0) y varianza (1), es un preproceso habitual y recomendable para muchas técnicas de aprendizaje automático. Por ello, consideraremos las series temporales normalizadas en el resto del proceso de modelización.

Notar que en el caso de las máquinas de vector soporte (SVMs, ver Sección 3.2) únicamente se centrarán las series. Esto se debe a la idea básica de las SVMs, que aplica funciones de transformación a los datos originales, generando un espacio con una dimensión mayor. En este proceso de "mapeo", muchas veces se produce una pérdida o distorsión de la normalización aplicada o de la escala de los valores que el usuario no puede cuantificar (Graf and Borer, 2001).

Figura 2.2: Distorsión de la normalización en el mapeo



Fuente: Graf and Borer (2001)

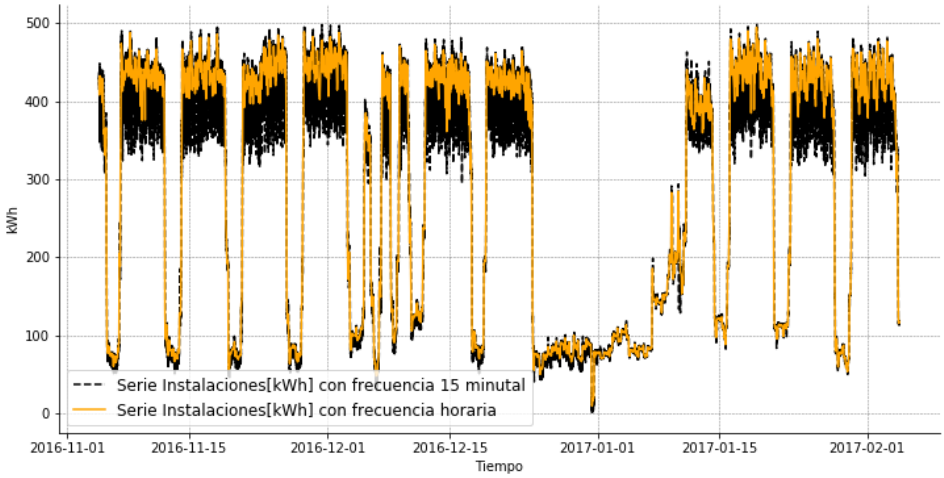
En este sentido, numerosos estudios presentan comparativas de metodologías de normalización para el caso de las SVMs analizando su efecto en el resultado final. Por ejemplo, Graf and Borer (2001) analiza la normalización de los inputs (datos de fotografías) en el espacio de entrada observando los problemas antes expuestos además

de, en última instancia, explorar los resultados de la normalización de la función del kernel, con correcciones en la posición del hiperplano de separación óptima. Por otro lado, Crone et al. (2006) comprueba distintos intervalos de normalización en el contexto de la predicción con SVMs y redes neuronales.

2.2.3. Visualización

En esta sección se llevará a cabo una breve visualización de los datos que permita observar el cambio sufrido por el conjunto de datos con el cambio de frecuencia temporal. La figura 2.3 muestra este cambio sufrido por la serie instalaciones una vez reducida la frecuencia temporal de 15 minutos a horaria. En ella, se observa como el principal efecto de esta reducción de frecuencia es la suavización de la serie suprimiendo varios de los outliers existentes previamente.

Figura 2.3: Comparación de frecuencias temporales



Fuente: Elaboración propia

Numéricamente, los estadísticos de las series también son una herramienta para comprobar esta transformación. Estos estadísticos para las series con frecuencia de 15 minutos se muestran en la tabla 2.3.

Tabla 2.3: Estadísticos con frecuencia de 15 minutos

	Pot.	GDir.	Tec.	Cal.	Cli.	Fr.	
Count	70975.00	70975.00	70975.00	70975.00	70975.00	70975.00	
Mean	0.88	60.48	63.77	79.99	33.76	40.08	
Std	1.30	70.63	82.45	89.38	24.26	77.86	
Min	0.00	0.00	0.00	0.00	0.00	0.00	
25 %	0.00	0.00	9.00	21.00	3.00	0.00	
50 %	1.00	0.00	27.00	48.00	35.00	1.00	
75 %	1.00	141.48	91.00	114.00	61.00	1.00	
Max	25.00	282.96	681.00	773.00	126.05	276.00	
	Al.	**Ins.**	**Prod 1**	**Prod 2**	**Prod 3**	**Tint.**	**Text.**
Count	70975.00	70975.00	70975.00	70975.00	70975.00	70975.00	70975.00
Mean	68.54	292.41	5.61	4.53	2.85	22.83	17.05
Std	24.87	139.67	6.23	6.03	3.81	2.29	7.59
Min	0.00	0.00	0.00	0.00	0.00	12.70	-1.80
25 %	69.00	140.35	0.00	0.00	0.00	21.90	11.00
50 %	80.00	337.58	3.00	0.00	0.00	22.50	16.90
75 %	84.00	408.19	11.00	10.00	6.00	23.45	22.70
Max	153.00	884.49	60.00	59.00	58.00	31.60	39.50

Fuente: Elaboración propia

Mientras que, con la frecuencia horaria los principales cambios que podemos observar son la evolución de las series hacia una tipología más suave así como la supresión de varios outliers, sobretodo en las series de producción. Por lo demás, cabe destacar que la serie instalaciones ya no encuentra su mínimo en el 0 y el número de observaciones se reduce significativamente, como se muestra en la tabla 2.4.

Tabla 2.4: Estadísticos con frecuencia horaria

	Pot.	GDir.	Tec.	Cal.	Cli.	Fr.	
Count	17745.00	17745.00	17745.00	17745.00	17745.00	17745.00	
Mean	0.81	60.43	63.72	79.85	33.74	40.09	
Std	1.06	69.27	82.27	88.73	24.22	77.57	
Min	0.00	0.00	0.00	0.00	0.00	0.00	
25 %	0.00	0.00	9.50	21.50	3.00	0.50	
50 %	0.50	0.00	27.00	48.50	35.00	1.00	
75 %	1.00	141.48	91.00	114.00	61.00	1.00	
Max	25.00	229.90	664.50	630.50	126.05	272.00	
	Al.	**Ins.**	**Prod 1**	**Prod 2**	**Prod 3**	**Tint.**	**Text.**
Count	17745.00	17745.00	17745.00	17745.00	17745.00	17745.00	17745.00
Mean	68.98	296.22	5.59	4.51	2.74	22.83	17.05
Std	24.55	140.57	5.41	5.44	3.09	2.29	7.58
Min	0.00	9.61	0.00	0.00	0.00	12.70	-1.55
25 %	71.00	141.09	0.00	0.00	0.00	21.90	10.99
50 %	80.50	348.88	6.00	0.00	1.00	22.50	16.90
75 %	83.50	411.80	10.50	9.50	5.50	23.45	22.70
Max	91.50	837.86	35.50	34.50	36.00	31.55	39.19

Fuente: Elaboración propia

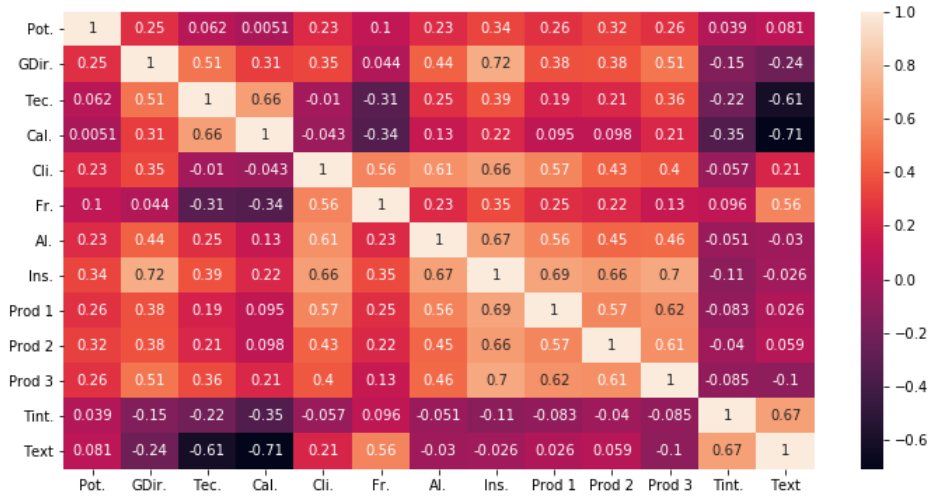
2.2.4. Correlación

Si bien para poder comparar los resultados con aquellos obtenidos por Meneses Agudo et al. (2019) los modelos predictivos se aplicarán únicamente a la variable Instalaciones, como se ha comentado anteriormente la consideración de variables dependientes o redundantes puede afectar e inestabilizar el modelo obtenido. En particular, este hecho será más determinantes para los métodos basado en GAM y/o SVM, los cuales

presentan una mayor sensibilidad de la predicción a los datos de entrada.

De forma descriptiva la figura 2.4 muestra la matriz de correlaciones cruzadas entre las variables del conjunto de datos. Observando los valores obtenidos por la variable “Instalaciones” las mayores dependencias se obtienen con las variables “Gas Directo”, “Climatizadores” y las tres líneas de producción.

Figura 2.4: Matriz de correlaciones



Fuente: Elaboración propia

CAPÍTULO 3

Metodología

Como se ha reflejado anteriormente, el presente trabajo se propone y construye como una continuación del trabajo efectuado por Meneses Agudo et al. (2019), extendiendo sus análisis no sólo a nuevos métodos si no a profundizar en diferentes aspectos del análisis. En el presente capítulo introducimos dichas extensiones, haciendo énfasis en los conceptos novedosos con respecto al trabajo anterior.

En primer lugar, al enmarcarse el problema de estudio en un análisis de series temporales cuyo objetivo es predecir el estado de un sistema dados los estados anteriores, se ha profundizado en la selección de variables en base a su significancia [1], tanto de las variables originales como de estados retrasados de las mismas (lags) con respecto a la variable objetivo a predecir. Notar que dicha significancia puede verse afectada por muchas cuestiones como la frecuencia de la serie, la estacionalidad o los efectos calendario.

Por otro lado se analizará la aplicabilidad de los Modelos Aditivos Generalizados (GAM, por sus siglas en inglés) en el ámbito de la predicción de series temporales, de forma que se puedan aislar los efectos de las componentes lineal y no lineal, y añadiendo, en última instancia, relaciones entre las variables. Esto permitirá inferir o concluir si para el caso en cuestión es adecuado efectuar un análisis con modelos no lineales (p.e. máquinas de vectores soporte (SVM) con núcleo no lineal), ya que podremos ver la aportación de cada componente.

Adicionalmente, se considerarán diferentes métodos basados en ensembles, incluyendo las principales aproximaciones de estas metodologías: Boosting, Bagging y Stacking, las cuales se describirán en este capítulo.

Por último, la aplicación de estos métodos se ha implementado en el lenguaje de programación de Python en su versión 3.7. Las librerías utilizadas serán “pyGAM” [2] para

[1] **Significancia:** término derivado de la existencia de una correlación significativa entre una variable y la objetivo.

[2] **pyGAM:** Generalized Additive Models in Python. Zenodo/DOI: 10.5281/zenodo.1208723

la confección del GAM, “LightGBM” [3] para el LightGBM y de “Scikit-Learn” [4] para el resto de métodos.

3.1. Modelos Aditivos Generalizados

Los Modelos Aditivos Generalizados (GAM), también conocidos como “wiggly models”, fueron propuestos por Hastie and Tibshirani (1986) y se entienden como una extensión de modelos más tradicionales como son la regresión lineal o los Modelos Lineales Generalizados (GLM). En cierta medida, “*permiten una mayor flexibilidad en el modelado de un predictor lineal por parte de un GLM como una suma de funciones generales para cada variable*” (Chouldechova and Hastie, 2015).

De tal manera, se puede considerar un GAM como una regresión aditiva de funciones suaves[5], cuya formulación inicial viene dada por:

$$g\left(\mu_i\right) = \mathbf{X}_i^* \boldsymbol{\theta} + f_1\left(x_{1i}\right) + f_2\left(x_{2i}\right) + f_3\left(x_{3i}\right) + \ldots, \quad i = 1, 2, , \ldots, n \tag{3.1}$$

Donde $\mathbf{X}_i^*$ representa el término constante, $\mu_i = E\left(Y_i\right)$, Y_i la variable respuesta, f_j para $j = 1, 2, 3$ las funciones suaves y x_{ji} las variables predictoras asociadas a cada función suave.

El valor añadido en esta regresión lo introducen los términos de la función f_j, que pueden aparecer de diversas formas, entre las que se encuentran términos lineales, de spline, de tipo factor y tensores o de interacción. Cada uno de estos términos trata de permitir un mejor ajuste, dotando a la regresión de gran flexibilidad de forma que, en última instancia, se pueda disponer de un ajuste distinto para cada predictor.

Mientras que la componente lineal hace referencia a un término clásico de una regresión lineal, las componentes tipo factor y tensores hacen referencia a la introducción de variables categóricas y de interacción entre sí, respectivamente. Por otro lado, el componente de spline [6] introduce la no linealidad en el problema y requiere del ajuste de una función base. En este punto, hay multitud de enfoques para su ajuste, siendo los más frecuentes los splines de regresión, suavizado, B-Splines (Splines base), P-Splines (splines con penalizaciones) y P-Splines cíclicos (splines con penalizaciones cíclicos), entre otros.

3.1.1. P-Splines

En esta memoria se emplearán funciones base de tipo P-spline, ya que son los que se encuentran implementados en pyGAM además de que, los B-splines, se pueden definir como un caso específico de los P-Splines con penalización igual a 0. Este concepto

[3] **Lightgbm:** https://lightgbm.readthedocs.io/en/latest/

[4] **Scikit-Learn:** https://scikit-learn.org/stable/

[5] **Función suave:** aquella con derivadas de todos los órdenes (Aguilar Barreiro, 2019)

[6] **Spline:** curva diferenciable y definida en partes a través de polinomios, donde se imponen restricciones en los puntos de unión, que se llamarán nodos.

deriva de las step-functions y de los splines de regresión y suavizado, que es necesario introducir previamente.

Las steps-functions emplean un enfoque que radica en la división del rango de los predictores en intervalos, ajustando a cada uno distintas constantes. Este enfoque es generalmente utilizado en la creación de variables dummies, que toman valor 1 en un intervalo concreto y 0 en los demás.

Una ampliación de las step-functions son los splines de regresión. Los splines de regresión tratan de ajustar un polinomio de grado m sobre los diferentes intervalos de un predictor X, en lugar de ajustar uno de grado d sobre todo el rango de este predictor, siendo $d > m$.

Los puntos que delimitan la separación entre intervalos se denominan nodos y las funciones aplicadas a los distintos intervalos se estiman a través del ajuste por mínimos cuadrados.

Por otro lado, los splines de suavizado son una herramienta que permite la representación de la evolución de una variable respuesta como una función de varios predictores lineales. En este caso no es posible el ajuste por mínimos cuadrados, por lo que O'Sullivan (1986) introdujo penalizaciones en la segunda derivada de la curva, de forma que:

$$RSS(f, \lambda) = \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_{x_1}^{x_n} f''(x)^2 dx \tag{3.2}$$

donde la primera componente representa la distancia entre los datos observados (y_i) y predichos por el modelo ($f(x_i)$), y la segunda la penalización a la curvatura de la función ($f''(x)$). Finalmente, λ representa el grado de penalización a esa curvatura de la función.

De estos dos últimos enfoques nacen los splines con penalizaciones (P-splines). Mientras que, por un lado, los splines de regresión presentan un método de estimación ampliamente conocido, se torna complicada la selección del número de nodos. Los splines de suavizado, por otro lado, utilizan tantos parámetros como observaciones introduciendo un problema de dimensionalidad.

Los P-Splines fueron introducidos por Eilers and Marx (1996) y "*combinan lo mejor de ambos enfoques: utilizan menos parámetros que los splines de suavizado, pero la selección de los nodos no es tan determinante como en los splines de regresión*" (Durbán, 2008). Esto ocurre debido a que la penalización que se introduce es discreta y, en última instancia, se aplica únicamente a los coeficientes reduciendo la dimensionalidad del problema.

En este caso, la metodología de estimación se basa en el uso de una base de función para la regresión y en la introducción de la penalización en la función de verosimilitud.

La metodología de estimación consiste en el uso de una función base para la regresión y en la introducción de la penalización a la función de mínimos cuadrados, dando lugar a la función de mínimos cuadrados penalizados.

La penalización presentará la siguiente formulación:

$$P = \lambda \sum_{j} \left(\Delta^d j \right)^2 \tag{3.3}$$

donde Δ^d es el operador de diferencias de orden d. Ahora, el nivel de suavizado de la curva no dependerá del número de nodos sino del parámetro de penalización λ.

Dado un problema de regresión estándar:

$$\boldsymbol{y} = \boldsymbol{B}\boldsymbol{a} + \boldsymbol{\epsilon} \tag{3.4}$$

siendo $\boldsymbol{B} = \boldsymbol{B}(\boldsymbol{x})$ una de las funciones base referida anteriormente dado x.

Uniendo 3.3 y 3.4 obtendremos la función de mínimos cuadrados penalizados, de donde se obtiene el siguiente problema de optimización:

$$\min(y - Ba)'(y - Ba) + \lambda a' D' D a \tag{3.5}$$

donde $P = \lambda D'D$ es la encargada de penalizar los coeficientes con λ como parámetro de suavizado.

Dando lugar a:

$$\hat{\boldsymbol{a}}_\lambda = \left(\boldsymbol{B}'\boldsymbol{B} + \boldsymbol{\lambda}\boldsymbol{D}'\boldsymbol{D} \right)^{-1} \boldsymbol{B}'\boldsymbol{y} \tag{3.6}$$

con los coeficientes ya estimados.

Cabe destacar que para valores grandes de lambda, es decir, $\lambda \to \infty$, los coeficientes se aproximarán a cero y el problema se vuelve un ajuste polinómico. En caso contrario, si $\lambda \to 0$ se estaría en un problema con B-Splines o de mínimos cuadrados ordinarios.

3.2. Máquinas de Vectores Soporte

Las máquinas de vectores soporte (SVMs) son algoritmos de aprendizaje supervisado que fueron desarrollados por Cortes and Vapnik (1995). Si bien, en primera instancia, las SVMs se diseñaron para la resolución de problemas de clasificación, recientemente se ha extendido su aplicación a problemas de regresión. En este contexto, la casuística en sendos problemas es bastante similar, pudiendo pasar de un problema de clasificación a uno de regresión introduciendo una función de pérdida.

Varios ejemplos de uso de las SVMs pueden ser el de Chen et al. (2004) introducido en la sección 1 para la predicción de carga eléctrica. En un contexto financiero, dos buenos ejemplos son los análisis de Huang and Wu (2008) y Huang et al. (2005), para sendos análisis de predicción sobre la evolución de diferentes índices bursátiles en Asia.

Por otro lado, en el caso de Smola and Schölkopf (2004) se realiza un repaso a las ideas que subyacen a las máquinas de vectores soporte y los distintos algoritmos empleados en su entrenamiento. Además, Crone et al. (2006) evalúa el rendimiento tanto de las SVMs como de distintos tipos de redes neuronales en el ámbito de las series temporales, a partir de distintos criterios de preprocesamiento de los datos.

La solución de un problema de regresión utilizando SVMs e introduciendo funciones de kernel sería la siguiente:

$$f(\mathbf{x}) = b + \mathbf{w}^T\mathbf{x} = b + \sum_{i=1}^{n} \alpha_i k\left(\mathbf{x}_i, \mathbf{x}\right) \tag{3.7}$$

donde b es el término constante, α_i el factor de ponderación de la función de kernel. Además, $k\left(x_i, x\right)$ representa un ejemplo de una familia de funciones que se denominan funciones de kernel. "*Existen multitud de funciones kernel y, generalmente, todas ellas se basan en el cálculo de productos internos entre dos vectores*"(Fletcher, 2009).

En esta memoria, el uso de las SMVs se elabora a través de la librería de Python de Scikit-Learn, donde las funciones de kernel más utilizadas son las siguientes:

Tabla 3.1: Funciones de kernel

	Función de kernel
Lineal	$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \mathbf{x}_i^T\mathbf{x}_j$
Radial Basis Function	$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = e^{-\left(\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{2\sigma^2}\right)}$
Polynomial	$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left(\mathbf{x}_i \cdot \mathbf{x}_j + a\right)^b$

Fuente: Elaboración propia

Una de las claves de las máquinas de vectores soporte es el "kernel trick", que se encuentra implícito en la ecuación 3.7. Consiste en que, si las funciones de kernel pueden ser proyectadas a un espacio no lineal a través de una potencial "función de proyección" no lineal $\mathbf{x} \longmapsto \phi(\mathbf{x})$, solo se necesitarían conocer los productos de los inputs en el espacio de características, sin necesitar conocer ϕ.

Las SVMs se centran, por tanto, en la proyección del espacio vectorial de los predictores x_t en uno de mayor dimensión, que está formado por transformaciones no lineales de dichos predictores, cuyo producto escalar se aproxima o modeliza por esa función $k(\cdot, \cdot)$.

El procedimiento de estimación se resume en el siguiente problema de optimización, siendo k la matriz de kernel compuesta de los elementos $k\left(\mathbf{x}_i, \mathbf{x}_j\right)$:

$$\begin{aligned} \text{máx}_{\alpha} \quad & \mathbf{y}^T\boldsymbol{\alpha} - \epsilon|\boldsymbol{\alpha}| - \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} \\ \text{s.t.} \quad & \boldsymbol{\alpha}^T\mathbf{1} = 0 \\ & |\alpha_i| \leq C, \forall i \end{aligned} \tag{3.8}$$

donde ϵ representa el parámetro de la función de pérdida y $\leq C$ el de regularización. Resolviendo este problema se obtendría 3.7.

3.3. Métodos basados en ensembles

Tanto en el trabajo de Meneses Agudo et al. (2019) como en las secciones anteriores, las soluciones se han planteado desde el prisma de obtener un único mejor modelo de predicción. Una alternativa a esta aproximación son los métodos basados

en ensembles, los cuales se fundamentan en que la combinación de muchos modelos puede aproximar mejor, y/o de forma más robusta, la variable objetivo que cada uno de forma individual. Dentro de esta familia de modelos podemos distinguir tres ramas principales denominadas: Bagging, Boosting y Stacking, considerando las dos primeras modelos homogéneos (p.e. regresión lineal) mientras que el tercero considera modelos heterogéneos (p.e. árboles de decisión y regresión). En nuestro caso, los modelos considerados en el caso de Bagging y Boosting serán los árboles de clasificación y regresión. De este modo podremos ampliar y, al menos hasta cierto punto, completar la batería de modelos desarrollados en el trabajo de Meneses Agudo et al. (2019).

3.3.1. Boosting

La técnica de boosting, que fue desarrollada por Freund (1995), está basada en la confección de multitud de "weak learners" homogéneos encadenados iterativamente, de forma que se termine alcanzando un estimador robusto tras un número finito de iteraciones. Es decir, a medida que se van entrenando "weak learners", estos se van añadiendo al modelo de ensembles hasta componer el modelo final ("strong learner").

El algoritmo AdaBoost, introducido por Freund et al. (1996), fue el primero derivado de esta técnica y empleado para problemas de aprendizaje supervisado. Su funcionamiento es sencillo, en una primera iteración todas las muestras obtienen los mismos pesos o ponderaciones. Sin embargo, tras la primera aproximación obtenida se asignan mayores pesos a las muestras mal clasificadas, proceso que se repite sucesivamente hasta haber obtenido un estimador lo suficientemente preciso.

Un cambio en la aproximación del algoritmo AdaBoost fueron los métodos Gradient Boosting' introducidos por Friedman (2001). En ambos casos, el modelo de ensembles presenta la siguiente forma:

$$s_L(.) = \sum_{l=1}^{L} c_l \times w_l(.) \tag{3.9}$$

donde c_l representa los coeficientes y w_l los "weak learners".

La diferencia más evidente se encuentra dentro del proceso iterativo necesario para encontrar el modelo óptimo. De esta forma, Gradient Boosting induce el problema hacia uno en el que se emplea la técnica del descenso del gradiente para la optimización.

El proceso de optimización en AdaBoost se puede representar:

$$s_l(.) = s_{l-1}(.) + c_l \times w_l(.) \tag{3.10}$$

donde c_l representa los coeficientes y w_l los "weak learners", mientras que $s_{l-1}(.)$ representa la mejora entre "weak learners".

El último paso será la actualización de c_l y w_l:

$$(c_l, w_l(\cdot)) = \arg\min E\left(s_{l-1}(\cdot) + c \times w(\cdot)\right) \tag{3.11}$$

En Gradient Boosting:

$$s_l(.) = s_{l-1}(.) - c_l \times \nabla_{s_{l-1}} E\left(s_{l-1}\right)(.) \tag{3.12}$$

donde c_l representa los coeficientes, $E(.)$ el error de ajuste y $-\nabla_{s_{l-1}} E\left(s_{l-1}\right)(.)$ el opuesto del gradiente del error de ajuste en el modelo $l-1$, los pseudo-residuos.

Se puede observar que, mientras que AdaBoost resuelve las optimizaciones de forma local, Gradient Boosting emplea la técnica de descenso del gradiente, aplicable a multitud de funciones de pérdida. Por esto, podría considerarse a Gradient Boosting como una generalización de AdaBoost.

Implementaciones

La optimización del algoritmo Gradient Boosting trajo consigo varias implementaciones que serán utilizadas en esta memoria.

La primera de ellas es el Extreme Gradient Boosting (XGBoost), introducido por Chen and Guestrin (2016) como una implementación específica de Gradient Boosting. En este sentido, se utilizan aproximaciones más precisas a la hora de buscar la confección óptima de cada árbol.

El cambio más importante se encuentra dentro del proceso de optimización, donde se pasan a utilizar las derivadas de segundo orden a la hora de buscar la dirección del gradiente, lo que permite acelerar este proceso. Por otro lado, también se actualiza la forma de regularización con las normas L1 y L2, que permiten aumentar la capacidad de generalización del modelo.

Otro de los algoritmos que se van a utilizar es el Light Gradient Boosting Machine (LightGBM), su formulación queda recogida por Ke et al. (2017). El enfoque de LightGBM es similar a los descritos anteriormente, la novedad reside en la introducción del nuevo método de muestreo GOSS (Gradient-based One-Side Sampling).

Esta técnica parte de la suposición de que las instancias de la muestra que poseen gradientes pequeños ya se encuentran entrenados, dejándolos de lado y seleccionando aquellas instancias con gradientes grandes, que se consideran sub-entrenados. Por esto, la técnica GOSS se entiende como una técnica para la reducción de la muestra, mejorando el tiempo de cómputo con respecto a las primeras implementaciones.

La última implementación es Catboost, desarrollado por Dorogush et al. (2018), que tiene dos importantes innovaciones con respecto a sus antecesores. Por un lado, implementa el Boosting "ordenado" de forma alternativa al enfoque de permutación anterior a la hora de confeccionar los árboles mientras que, por otro, introduce otro algoritmo innovador a la hora del tratamiento de predictores categóricos.

3.3.2. Bagging: Random Forest

Los modelos de Random Forest, se enmarcan dentro de los métodos de ensemble y son una vertiente de análisis nacida de los procesos de Bagging. En ellos, se tratan de

combinar multitud de árboles profundos que permiten producir resultados con menores varianzas. A la hora de desarrollar cada uno de los múltiples árboles que conforman el random forest, se lleva a cabo un muestreo tanto sobre el conjunto de datos como sobre los regresores, seleccionándolas aleatoriamente y permitiendo reducir las correlaciones entre árboles.

El algoritmo sigue las siguientes fases:

Tabla 3.2: Algoritmo Random Forest

Algoritmo Random Forest
1-Separación muestras en entrenamiento/validación/test.
2- Construcción random forest:
a. Selección aleatoria de los datos y regresores de entrenamiento
b. Con esta muestra se entrena el árbol.
c. Se repiten a y b tantas veces como árboles haya.
d. Se obtiene el random forest como promedio.
3- Predicción sobre el test.

Fuente: Elaboración propia

De esta forma, el random forest se puede definir como:

$$s_L(\cdot) = \frac{1}{L}\sum_{l=1}^{L} w_l(.) \tag{3.13}$$

donde w_l representa los "weak learners".

3.3.3. Stacking

Los métodos de Stacking fueron desarrollados por Wolpert (1992) y, a diferencia de los métodos Boosting y Bagging, estos métodos combinan "weak learners" heterogéneos para obtener uno más fuerte a partir de lo que se denomina un meta-modelo.[7]

Tabla 3.3: Algoritmo Stacking

Algoritmo Stacking
1-Separación muestras en entrenamiento/validación/test.
2- Construcción Stacking:
a. Entrenamiento y ajuste de los "weak learners" heterogéneos de primer nivel.
b. Predicción sobre la muestra de validación.
c. Entrenamiento del meta-modelo con las predicciones de los "weak-learners".
3- Predicción sobre el test.

Fuente: Elaboración propia

En este caso, los modelos se dividen en dos "niveles". Por un lado, los modelos del primer nivel serán aquellos "weak learners" que se hayan entrenado con los métodos anteriormente expuestos. En el segundo nivel, se entrenará un meta-modelo con las predicciones de los modelos del primer nivel, que dará lugar a la predicciones finales. Este

[7] **Meta-modelo**: modelo de segundo nivel entrenado con las predicciones de los modelos de primer nivel.

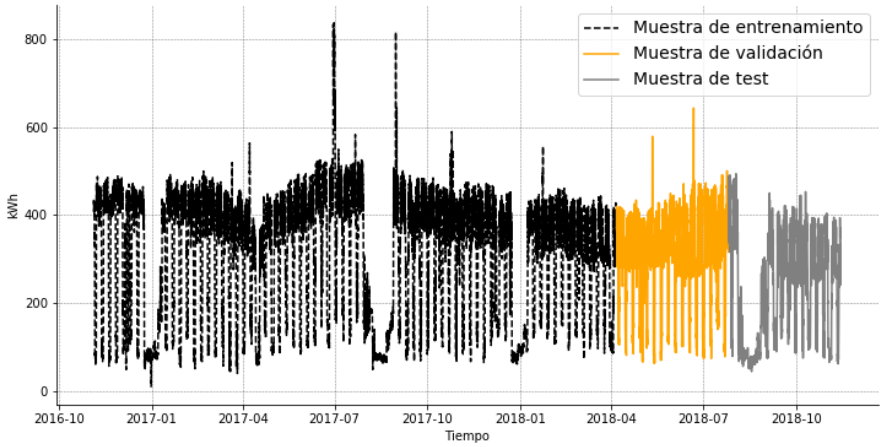
meta-modelo será de nuevo un GAM, con una implementación similar a la presentada en la sección 3.1.

Por su parte, para que esta técnica tenga éxito y consiga reducir el error de test de los modelos del primer nivel donde, según Hansen and Salamon (1990), es necesario que estos tengan un buen rendimiento y sean lo suficientemente diversos.

3.4. División Entrenamiento/Validación/Test

La división del conjunto de datos se efectuará en línea a la propuesta por Meneses Agudo et al. (2019). Por tanto, se hará una división de la muestra donde se entrenará el modelo con el 70 % del conjunto de datos mientras que para la validación y test se tomará el 15 % en cada una.

Figura 3.1: División Entrenamiento/Validación/Test



Fuente: Elaboración propia

En esta memoria se implementará la predicción en bloque[8] evaluada para distintos horizontes temporales (t y $t+2$), con el objetivo de probar distintos tiempos para la toma de decisiones operativa, que será distinta en función del ámbito de aplicación.

Para analizar las diferencias de ambas aproximaciones, en la sección 4 se considerará la predicción sobre t (a corto plazo), permitiendo la comparación con la memoria de Meneses Agudo et al. (2019) y, en segundo lugar, la predicción sobre $t+2$ (a medio plazo) permitiendo una comparación entre ambas metodologías.

3.5. Errores de predicción y métricas

Se puede definir el error de predicción como la diferencia entre la observación real y

[8] **Predicción en bloque**: Predicción sobre un número definido de instantes futuros donde el modelo trabaja únicamente con datos observados según van siendo necesarios y no con predichos.

la predicha por el modelo, de forma que:

$$e_t = y_t - \hat{y}_t \tag{3.14}$$

donde y_t representa el valor observado y $\hat{y}_t$ el predicho.

Utilizaremos distintas métricas, que son fórmulas matemáticas capaces de darnos una intuición, no siempre exacta, de lo bueno que es nuestro modelo y permitirnos evaluarlo correctamente. Es decir, otorgarán una intuición de si las predicciones se encuentran más o menos cerca de las observaciones reales.

Encontrándose en un problema de regresión, los outputs que se obtendrán del algoritmo serán valores numéricos, donde el abanico de métricas se vuelve bastante amplio, quedando a expensas del problema y a elección del usuario la utilización del más adecuado.

Varias de las métricas más utilizadas en casos de regresión son el Error Medio Absoluto (EMA) y la Raíz del Error Cuadrático Medio (RECM), que destacan por ser fáciles de calcular e interpretar y que, cuantitativamente, son equivalentes.

$$MAE = mean(|e_t|) \tag{3.15}$$

$$RMSE = \sqrt{\text{mean}\left(e_t^2\right)} \tag{3.16}$$

donde ambos indicadores se calculan en base a 4.3.

Sin embargo, son medidas cuantitativas por lo que, en función de la serie que se analice, se pueden tener problemas de unidades. Para solventar estos problemas surgen otras métricas que dan los errores como porcentaje.

Una de ellas es el Porcentaje del Error Medio Absoluto (MAPE, por sus siglas en inglés), cuya formulación es la siguiente:

$$\text{MAPE} = \text{mean}\left(|p_t|\right) \tag{3.17}$$

donde $p_t = 100e_t/y_t$.

El problema de esta métrica es que toma valores absolutos cuando las predicciones toman valores nulos o próximos a cero. Además, la literatura establece que se penalizan más fuertemente los errores con valor negativo que positivo (Hyndman and Koehler, 2006). Para solventar esto, surge el Porcentaje del Error Medio Absoluto Simétrico (sMAPE), que se puede representar como:

$$\text{sMAPE} = \text{mean}\left(200\left|y_t - \hat{y}_t\right| / \left(y_t + \hat{y}_t\right)\right) \tag{3.18}$$

Sin embargo, el sMAPE no termina de corregir de forma contundente estos errores ya que penaliza más las predicciones que superan el valor real de las que se mantienen por debajo además de ser menos interpretable que el MAPE.

Dentro de la literatura en el contexto de métricas para la evaluación de modelos, hay estudios que no recomiendan el uso de sMAPE, como la publicación sobre métricas

para cuestiones de predicción por Hyndman and Koehler (2006). Siguiendo su análisis y recomendaciones, en esta memoria se empleará una medida de errores escalados, como es el Error Medio Absoluto Escalado (MASE):

$$\mathrm{MASE} = \mathrm{mean}\left(|q_t|\right) \tag{3.19}$$

donde

$$q_t = \frac{e_t}{\frac{1}{n-1}\sum_{i=2}^{n}|Y_i - Y_{i-1}|}$$

siendo Y_i e Y_{i-1} las observaciones de la variables dependiente.

Esta métrica produce un error de escalado menor que 1 si la predicción es mejor que la de persistencia [9] mientras que será mayor que 1 en caso contrario. Es decir, la obtención de un MASE menor que la unidad denota una predicción mejor que la de persistencia, mientas que si su valor es mayor que la unidad, la predicción será peor que esta persistencia.

Por otro lado, atendiendo a cuestiones de consistencia con la memoria de Meneses Agudo et al. (2019), se utilizará la métrica de sMAPE incluyendo la métrica MASE como valor añadido de esta memoria.

[9] **Persistencia:** predicción resultante del uso del la observación del instante $t-1$ como valor predicho en el instante t.

CAPÍTULO 4

Resultados

En esta sección se describirán los resultados obtenidos a partir de la aplicación de la metodología descrita en el capítulo 3. En primer lugar, se establecerá la configuración de cada método, tanto en la selección de predictores y del número de retardos como la optimización de sus parámetros. Después, se mostrarán los resultados de la predicción en el corto y medio plazo para, en última instancia, introducir la configuración del metamodelo, permitiendo así su comparación tanto con el resto de métodos como con el trabajo desarrollado por Meneses Agudo et al. (2019).

4.1. Configuración de métodos

Como se ha descrito en capítulos anteriores, el problema inicial a resolver en la configuración de los diferentes métodos es la selección de variables predictoras, el número de retardos a incluir en la regresión y el tamaño de la ventana deslizante. Notar que en el caso del GAM existe otra dimensión a configurar dada por el número y tipo de splines.

Dicha selección se realizará en base a los resultados obtenidos sobre la muestra de test, de forma que se considerará la configuración que minimice el error de esta muestra dado por el sMAPE, evadiendo así el sobreajuste. Dado que los splines afectan únicamente al GAM, el ajuste de la configuración se realizará de forma secuencial. En primer lugar se obtendrá el número de retardos, paso que será común a todos los métodos, y a continuación se ajustarán el número de splines considerados en el caso de GAM. De este modo, en la memoria se describirá en detalle esta etapa únicamente para el GAM, que engloba el proceso completo, utilizando en secciones subsiguientes la correspondiente configuración óptima obtenida para cada uno de los métodos a través de la misma metodología.

4.1.1. Extracción y selección de variables

La extracción de variables y su posterior selección tiene como objetivo eludir la po-

sibilidad de aparición de un retraso en la predicción. Por esto, se han creado nuevos predictores que se añadirán a los retardos y se calcularán sobre los mismos, de forma que se seleccionen los que proporcionen un valor añadido a las predicciones de cada modelo.

Los nuevos predictores a añadir a los retardos son los siguientes:

Tabla 4.1: Nuevos predictores

Variable	**Descripción**
Media móvil	Media de las series cada dos observaciones consecutivas, denominado ventana deslizante.
Tasa de crecimiento	Mide la variación entre dos observaciones consecutivas.
Tasa de crecimiento de 2º orden	Mide la variación entre la tasa de variación dos observaciones consecutivas.
Desviación estándar	Mide la dispersión de la serie.
Cuantil 0.75	Con una ventana deslizante de 5 observaciones, crea una serie seleccionando el valor que deja el 75 % de los mismos por debajo.
Cuantil 0.95	Con la misma ventana deslizante que en el caso anterior, crea una serie seleccionando el valor que deja el 95 % de los mismos por debajo.
Dummies temporales	Toman valores de 0 ó 1 indicando la existencia de efectos categóricos, parametrizando así la fecha. En este punto, se han añadido la hora del día, día de la semana, semana del año, trimestre del año y, por último, una que indica si la serie se encuentra o no en fin de semana.

Fuente: Elaboración propia

Una vez elaborada la extracción de variables, la selección del conjunto de las mismas que entrarán en cada modelo se efectuará minimizando el error cometido por este.

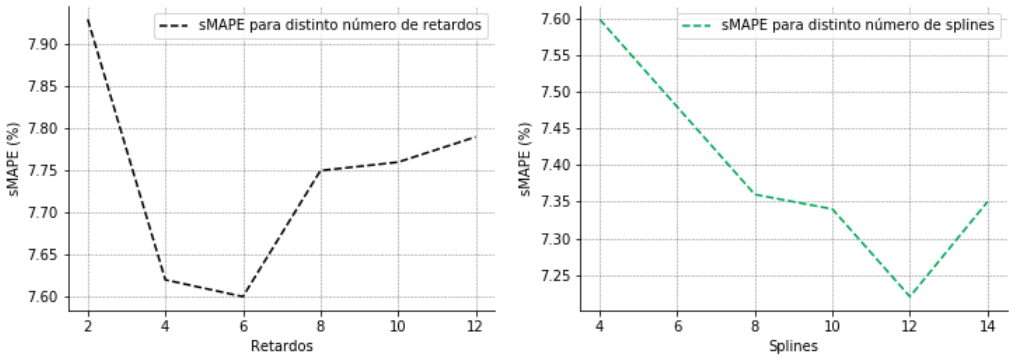
4.1.2. Modelo Aditivo Generalizado

En primer lugar se efectúa la optimización del número de retardos a emplear en la regresión. La forma intuitiva de proceder es evaluando los coeficientes dados por las funciones de de auto-correlación absoluta y parcial (ACF y PACF) de la serie. Sin embargo, la alta frecuencia de la serie temporal hace que existan una gran cantidad de retardos con correlaciones estadísticamente significativas, por lo que este apartado se encuentra condicionado, en última instancia, a la capacidad computacional disponible dado el coste de cómputo de este modelo.

En este sentido, se ha desarrollado la prueba con distintas cantidades de retardos de la variable objetivo. Una vez fijado éste se ha analizado su acierto para distintos números de splines, como se muestra en la figura 4.1.

Por tanto, primeramente se ha obtenido un número de retardos óptimo de 6 para, fijando estos, obtener un óptimo de 12 splines. Se ha confeccionado un GAM con splines de tercer orden cuya penalización se corresponde con la de segunda derivada introducida por O'Sullivan (1986). Los términos de la regresión se corresponden en su totalidad con términos de spline y su distribución será la Gaussiana Inversa que, al igual que para los GLM, pertenece a la familia exponencial y permite explicar no linealidades en los

Figura 4.1: Optimización de retardos y splines



Fuente: Elaboración propia

datos.

El hecho de no poder confeccionar un GAM incluyendo distintos tipos de términos se debe a la cantidad de predictores (383) incluidos en el modelo tras la selección de variables. Esta selección, como se ha comentado anteriormente, incluye 6 retardos y elimina la variable potable de la regresión. Además, de las variables descritas en la sección 4.1.1, en el GAM se han usado la media móvil y las dummies temporales descritas en la Tabla 4.1.

Más adelante, el modelo de ensembles presentará un GAM como meta-modelo en el que únicamente se tendrán 6 predictores, permitiendo una mayor libertad en su configuración.

4.1.3. Otros métodos

Support Vector Machine

Aplicando una optimización y selección de variables similar a la implementada anteriormente con el GAM, en la SVM con núcleo no lineal se ha obtenido una configuración óptima de 2 retardos aplicado a las series de Instalaciones, Gas Directo, Climatizadores y las producciones. De este proceso derivan la selección, en este caso, de las dummies temporales, media móvil y las series de cuantiles.

Por otra lado, el resultado de la optimización de parámetros para el problema de optimización definido en la ecuación 3.8 ha sido $C = 1.0$, $\epsilon = 0.05$ y $\alpha = 0.009$, dando lugar a la configuración que utilizaremos para este método.

Random Forest, XGBoost, LightGBM y CatBoost

La presentación de las configuraciones de los métodos de bagging y boosting, junto con sus implementaciones referidas en las secciones 3.3.2 y 3.3.1 se realizará de forma conjunta ya que comparten muchos de sus parámetros.

Los que se han ajustado son el número de árboles que componen el bosque (n_estimators o iterations), el número de muestras que se requieren para la división de un nodo interno (min_samples_split), el número de muestras requeridas en cada nodo terminal (min_samples_leaf), la profundidad del árbol (max_depth o depth), el número de predictores considerados para efectuar las divisiones (max_features), la tasa de aprendizaje (learning_rate), la fracción de muestras que se utilizan para ajustar los "weak learners" (subsample), la fracción mínima ponderada del total de pesos que debe estar en un nodo terminal (min_child_weight) y el número de rondas en las que el algoritmo se detendrá si no mejora una métrica objetivo, en nuestro caso el MAPE (early_stopping_rounds).

Las implementaciones de estos métodos de boosting y bagging presentan una gran capacidad de cómputo por lo que se ha podido incrementar notablemente el número de retardos en la regresión. Esto ha permitido comprobar la existencia de una periodicidad diaria clara, obteniendo un número óptimo de 24 retardos para todos los métodos. Por su parte, la selección de variables también incluye la mayoría de las propuestas en casi todos los casos.

En el caso de Random forest (bagging) la selección de variables mantiene las series de media móvil, dummies temporales, desviación estándar y las tasas de crecimiento de primer y segundo orden. Este es un caso parecido al de CatBoost, donde también la selección añade los cuantiles 0,75 y 0,95 a la regresión.

La selección arroja los mismos resultados para los modelos LightGBM y XGBoost, añadiendo únicamente la media móvil, las dummies temporales y la tasa de crecimiento de primer orden.

La parametrización de cada modelo responde a la siguiente tabla:

Tabla 4.2: Parámetros métodos boosting y bagging

Random Forest	**XGBoost**	**LightGBM**	**CatBoost**
n_estimators = 100	n_estimators = 500	n_estimators = 365	Depth = 8
min_samples_split = 10	Learning Rate = 0.07	Learning Rate = 0.078	Learning Rate = 0.05
min_samples_leaf = 10	min_child_weight = 4	min_child_weight = 4	Iterations = 700
max_depth = 20	max_depth = 5	max_depth = 6	early_stopping_rounds = 200
max_features = "auto"	Subsample = 0.70	Subsample = 0.70	Subsample = 0.70

Fuente: Elaboración propia

Esto permite una comparativa de configuración de cada modelo. Se puede comprobar que Random Forest necesita menos árboles para llegar a la solución que los demás y, sin embargo, es con diferencia el modelo más lento (Jhaveri et al., 2019). También destaca la profundidad de árbol de este método, mucho mayor que en los de boosting (Bauer and Kohavi, 1999).

Por último, puede verse como los modelos de boosting utilizan fracciones de muestras similares a la hora de establecer el ajuste de sus "weak learners" así como tasas de aprendizaje muy próximas. Donde hay más diferencias es en el número de árboles que componen cada método, donde CatBoost es el que más utiliza llegando a doblar a LightGBM. Esto puede deberse a la existencia de una relación positiva entre el número de árboles y el error de predicción de los modelos boosting. CatBoost presenta rangos

de creación más rápida de árboles (Dorogush et al., 2018), por lo que es viable entrenar más árboles sin emplear más tiempo que en el resto de métodos.

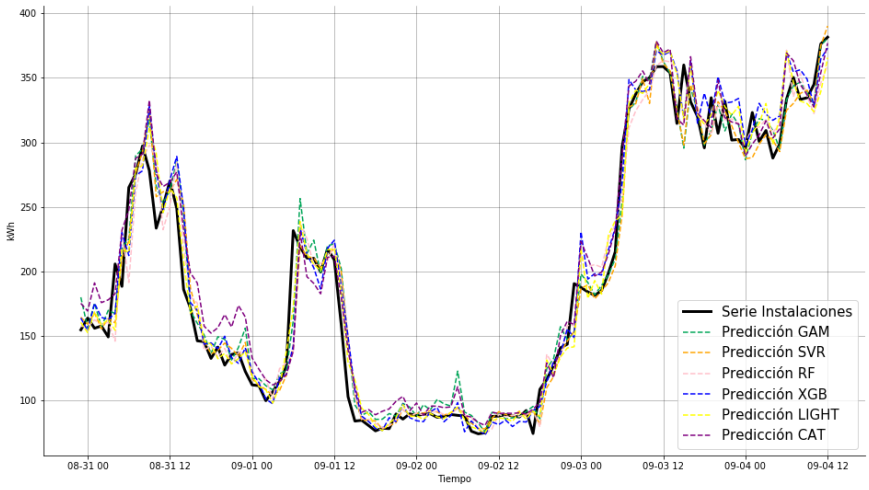
4.2. Predicciones

En el siguiente apartado, una vez establecida la configuración de los métodos en términos de características y parámetros, se introducirán los resultados obtenidos para las predicciones tanto a corto plazo (t) como a medio plazo ($t + 2$). Previamente, se observarán cómo se desenvuelven los métodos en comparación a los valores observados de la serie de instalaciones para después constatarlo con las métricas propuestas.

Predicción a corto plazo

La predicción a corto plazo de los métodos presenta la siguiente forma:

Figura 4.2: Predicción a corto plazo



Fuente: Elaboración propia

En la gráfica anterior se representan las predicciones para el periodo comprendido entre el 31 de Agosto y el 4 de Septiembre de 2018, donde se pueden intuir ligeras diferencias de comportamiento en la predicción según el método.

La tónica general es la observación de un pequeño “delay” o retraso en la predicción ya que, generalmente, los métodos no son capaces de anticiparse a los cambios bruscos de la serie y la predicción suele adelantarse a la serie real. Una muestra de este hecho puede verse en la incapacidad de las predicciones de anticipar los picos en la serie observada. Previsiblemente, este hecho se produce por un exceso de relevancia del primer retardo de la serie de instalaciones, por lo que el modelo tendría excesivamente en cuenta esta variable para elaborar su predicción, lo que le haría reaccionar tarde a cambios bruscos en el valor real.

Una forma de solucionar este problema podría ser la realización de una ingeniería de variables más extensiva para tratar de reducir el peso de este primer retardo en la capacidad de decisión del modelo.

Por otro lado, y atendiendo a cuestiones concretas de cada método, salta a la vista la tendencia de CatBoost a sobre estimar el valor real de predicción. Esto se observa fácilmente en las zonas bajas de la serie donde no suele llegar al nivel de la original, hecho que se observa también, aunque en menor medida, en los demás métodos. También se observa como LightGBM o SVM son capaces de reducir ligeramente el "delay" generalizado.

Sin embargo, estas primeras intuiciones corresponden a un segmento de la serie de test completa, por lo que las métricas de error introducidas en el capítulo 3 pueden otorgar una visión más global del comportamiento de cada método:

Tabla 4.3: Errores de predicción a corto plazo

Método	**sMAPE**	**MASE**
GAM	7,22 %	0,91
SVM	**5,86 %**	**0,79**
RF	6,63 %	0,88
XGB	6,30 %	0,83
LIGHT	6,32 %	0,83
CAT	6,9 %	0,80

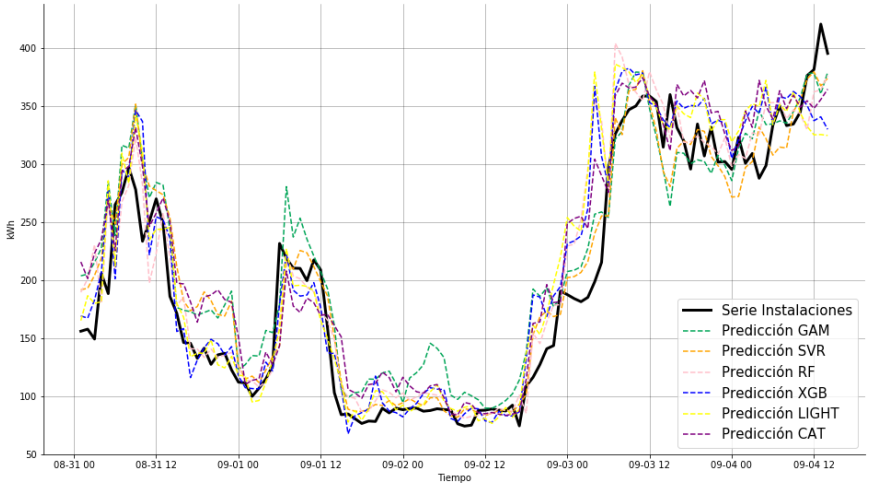
Fuente: Elaboración propia

La tabla 4.3 muestra que el método que mejores resultados de validación obtiene para ambas métricas en el formato de predicción a corto plazo es la SVM mientras que el GAM reporta el peor resultado, también en ambas métricas. Los métodos basados en árboles arrojan resultados muy similares entre si, aunque hay cambios en función de la métrica ya que según sMAPE CatBoost es el peor de estos pero según MASE es el mejor. Por su parte, si se observa el MASE se puede comprobar que todos los métodos mejoran la predicción de persistencia y, por tanto, aportan valor a la misma.

Predicción a medio plazo

Teniendo en cuenta una mayor horizonte temporal, la perspectiva y los resultados son distintos. En base a la descripción introducida en el capítulo anterior se ha elaborado la predicción sobre 3 observaciones futuras en el test, de modo que se establecería un tiempo suficiente para posibilitar la toma de determinadas decisiones operativas o la detección de averías, obteniendo los siguientes resultados:

Figura 4.3: Predicción a medio plazo



Fuente: Elaboración propia

La figura 4.3 muestra los resultados de predicción a medio plazo para los distintos métodos entre el entre el 31 de Agosto y el 4 de Septiembre de 2018, arrojando unos resultados de predicción con mayor grado de oscilación que en el caso del corto plazo. En general los métodos tienden a presentar un mayor retraso en la predicción que en el caso anterior y, aunque el modelo con menor sMAPE sigue siendo la SVM y el mayor el GAM, los modelos basados en árboles muestran resultados diferentes a los reflejados en el punto anterior (ver Tabla 4.3).

Los errores de predicción en este caso son:

Tabla 4.4: Errores de predicción a medio plazo

Método	**sMAPE**	**MASE**
GAM	13,79 %	1,64
SVM	**9,77 %**	**1,34**
RF	10,40 %	1,36
XGB	11,61 %	1,45
LIGHT	11,87 %	1,52
CAT	12,38 %	1,48

Fuente: Elaboración propia

En este caso, random forest mejora a LightGBM y XGBoost mientras que CatBoost sigue siendo el que peor resultado reporta de los mismos. Cabe destacar los valores obtenidos para el MASE, donde los métodos comienzan a otorgar perores resultados que la persistencia, lo que hace indicar que el método de entrenamiento debería ser distinto para este tipo de predicción.

Los aspectos remarcables se identifican con una mayor oscilación de las series dado que el modelo está más obligado a arriesgarse y otorgar un resultado sin ver varios de los anteriores. Además, no desaparece el ligero “delay” de la predicción. Realmente, los

errores de predicción aumentarán junto con el horizonte temporal que se establezca, sin embargo, en este caso un horizonte de 3 horas puede ser tiempo suficiente para que sea útil a la hora de prevenir accidentes laborales, detectar anomalías o ahorrar de costes.

4.3. Stacking

En la presente sección se va a pasar a la presentación del modelo stacking de ensembles introducido en el capítulo 3. Este se compondrá de dos niveles, el primero donde los inputs son las observaciones de la muestra de entrenamiento y de la que obtendremos los outputs en forma de predicción. Esas predicciones se efectuarán sobre la muestra de validación (ver Figura 3.1), con lo que se entrenará el meta-modelo en el segundo nivel y cuyo output serán las predicciones sobre la muestra de test. Para la construcción de este meta-modelo hemos optado por considerar un modelo GAM, que nos permita incluir términos describiendo comportamientos diferentes entre los predictores (linea, no lineal, interacciones, etc...).

Por tanto, se comentará la configuración del GAM utilizado como meta-modelo y se presentarán las predicciones sobre el test tanto en el formato de corto como de medio plazo. Por último, se establecerá una comparativa en términos de errores de predicción con la memoria de Meneses Agudo et al. (2019).

4.3.1. Configuración del meta-modelo

Como se introdujo en la sección 3.1, se ajustará un GAM como meta-modelo, que presenta la siguiente estructura:

Figura 4.4: Resumen GAM (meta-modelo)

GAM

Distribution:	InvGaussDist	Effective DoF:	58.5068
Link Function:	LogLink	Log Likelihood:	-3553773.7677
Number of Samples:	2654	AIC:	7107666.549
		AICc:	7107669.3256
		GCV:	0.0001
		Scale:	0.0001
		Pseudo R-Squared:	0.9857

Feature Function	Lambda	Rank	EDoF	P > x	Sig. Code
s(0)	[0.95]	21	15.0	1.11e-16	***
s(1)	[0.95]	21	10.9	7.77e-16	***
s(4)	[0.95]	21	9.4	1.11e-16	***
s(3)	[0.95]	21	10.1	1.11e-16	***
s(5)	[0.95]	21	6.1	1.11e-16	***
te(1, 2)	[0.95 0.95]	100	4.7	1.15e-04	***
te(0, 1)	[0.95 0.95]	100	0.6	5.86e-02	.
te(4, 5)	[0.95 0.95]	100	0.3	1.11e-16	***
te(4, 0)	[0.95 0.95]	100	0.8	1.11e-16	***
te(4, 2)	[0.95 0.95]	100	0.2	1.11e-16	***
te(0, 2)	[0.95 0.95]	100	0.3	1.11e-16	***
te(0, 4)	[0.95 0.95]	100	0.0	1.11e-16	***
te(0, 2)	[0.95 0.95]	100	0.0	1.11e-16	***
te(0, 5)	[0.95 0.95]	100	0.0	1.11e-16	***
te(2, 5)	[0.95 0.95]	100	0.1	1.11e-16	***
intercept		1	0.0	2.55e-14	***

Significance codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Fuente: Elaboración propia

En la figura 4.4 se pueden observar los términos incluidos en la regresión (Feature Function), su significación estadística (Sig. Code), el parámetro de regularización (Lambda) y el número de splines (Rank), entre otras.

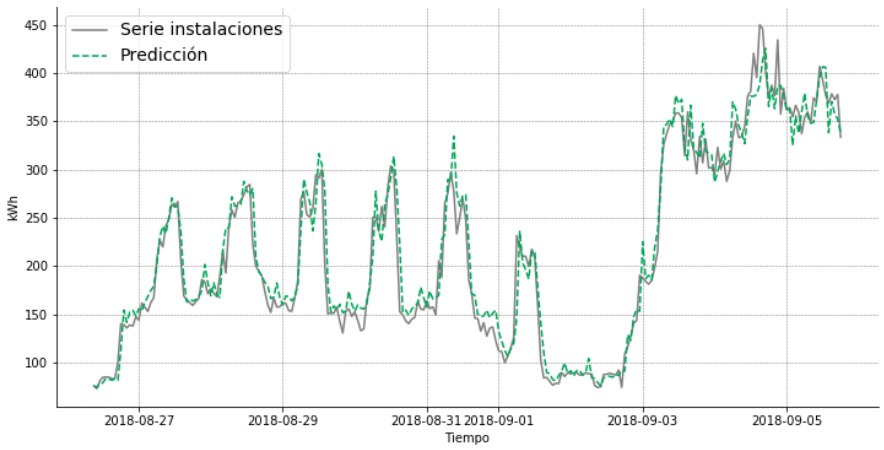
En la regresión no se han añadido términos lineales ni de tipo factor, ya que no se tienen variables categóricas, por lo que se corresponden con términos de spline (s()) y tensores (te() y to()), que introducen interacción entre inputs. Se ha añadido un término de spline por cada modelo de predicción, a excepción del XGBoost ya que empeoraba significativamente el resultado. Por tanto, los términos s(0), s(1), s(3), s(4) y s(5) se corresponden con GAM, SVM, Random Forest, LigthGBM y CatBoost, respectivamente. Las predicciones de XGBoost se añadirán en los términos tensores, donde se han interpuesto distintas y diversas interacciones entre todos los inputs del modelo. Por su parte, se han ajustado 21 splines para el caso de los términos de spline y 100 para los tensores.

El parámetro de regularización λ se ha fijado en 0.95, siendo todos los términos de la regresión significativos a excepción de la interacción impuesta sobre el GAM y la SVM (te(0,1)). La función de distribución será, como en el caso del modelo individual, la Gaussiana Inversa perteneciente a la familia exponencial.

4.3.2. Predicciones

Como para el caso de los métodos individuales, se ha probado el stacking en los formatos tanto de predicción a corto plazo como a medio plazo, cuyo objetivo es el de comprobar si la calidad de las predicciones aumenta con respecto a los modelos individuales.

Figura 4.5: Predicción a corto plazo



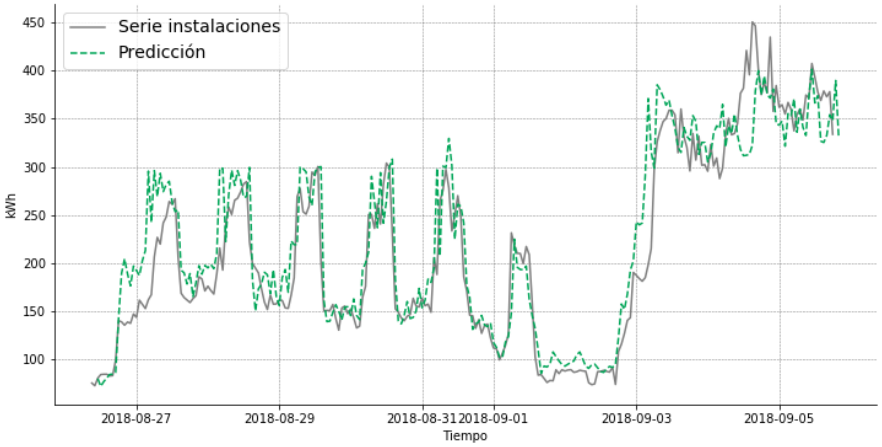
Fuente: Elaboración propia

Al igual que en la figura 4.2, en la figura 4.5 se muestran los resultados de predicción a corto plazo en comparación a la serie observada, para el periodo comprendido entre

el 27 de Agosto y el 6 de Septiembre de 2018. En ella se observa como el método de stacking, como un conjunto de métodos heterogéneos, proporciona un resultado que se encuentra dentro del rango presentado por los modelos de forma individual, lo que podría ser lo esperado. Sin embargo, el resultado es cuantitativamente mejor y consigue, en cierta medida, paliar la presencia del "delay" de predicción observado en los métodos individuales. Del mismo modo, si bien no consigue anticiparse en las zonas de picos y bajadas mas pronunciadas, el modelo es capaz de capturar la estacionalidad de la serie. A la vista de los resultados, en este formato de predicción a corto plazo representa una mejora con respecto a los métodos individuales, aunque tiene varias limitaciones que se comentarán posteriormente.

Por su parte, la figura 4.6 muestra los resultados de predicción a medio plazo de tres puntos del meta-modelo en el periodo comprendido entre el 27 de Agosto y el 6 de Septiembre de 2018.

Figura 4.6: Predicción a medio plazo



Fuente: Elaboración propia

En este caso la predicción se pierde con frecuencia y no se consigue, en muchas ocasiones, ajustar el nivel ni la volatilidad de la serie, donde se infiere que el modelo stacking no es capaz de mejorar a muchos de los modelos individuales. Este hecho puede verse también en las métricas de predicción:

Tabla 4.5: Errores de predicción Stacking

Método	sMAPE	MASE	Método Predicción
GAM (Meta-Modelo)	5,85 %	0,74	Corto plazo
GAM (Meta-Modelo)	11,76 %	1,51	Medio plazo

Fuente: Elaboración propia

El resultado de predicción del meta-modelo mejora a los métodos individuales en la predicción a corto plazo, mientras que, al mover el horizonte temporal, el resultado empeora significativamente.

Si bien, hablando en términos de predicción en un entorno industrial, este modelo quizá no sería el más adecuado para implementar a nivel operativo y, por tanto, adoptar nuevos enfoques para mejorar las predicciones a mayor horizonte surge del presente estudio como una posible línea futura de trabajo.

4.4. Comparativa

Por último, se compararán los resultados por los obtenidos por Meneses Agudo et al. (2019). La tabla 4.6 muestra los valores de sMAPE para todos los métodos implementados en ambas memorias:

Tabla 4.6: Comparativa de resultados

Actual		Meneses Agudo et al. (2019)	
Método	sMAPE	Método	sMAPE
GAM	7,22 %	Persistencia	8,00 %
SVM	5,86 %	R.Lineal	8,53 %
R.Forest	6,63 %	SVR Lineal	**7,45 %**
XGBoost	6,30 %	MLP	16,61 %
LightGBM	6,32 %	GRU	9,44 %
CatBoost	6,90 %	LSTM	9,01 %
Stacking	**5,85 %**	SARIMA	9,67 %

Fuente: Elaboración propia

La batería de modelos es extensa y engloba desde las metodologías clásicas de análisis de series temporales, como los modelos SARIMA, hasta métodos basados en árboles implementados en la presente memoria, pasando por modelos de deep learning.

Los resultados obtenidos en ambas memorias reflejan unos errores de predicción bastante aceptables con respecto a las series temporales que se están tratando, en el contexto de datos SCADA y son ya cuantitativamente poco mejorables, al menos en el caso de predicción a corto plazo.

Pasando a la comparación de métodos, se observa que una extensa parametrización y el hecho de no poseer una base de datos excesivamente grande permite a las SVM otorgar un resultado bastante decente. Este hecho puede ser el que frene a los modelos de deep learning que, por su parte, necesitan de mayor cantidad de datos para entrenarse correctamente. Los modelos basados en árboles también suponen una buena solución al problema de una forma bastante eficiente mientras que, por otro lado, la regresión lineal y los GAM que pueden ser soluciones comparables producen resultados que no se alejan en excesivo de métodos más complejos para este conjunto de datos. El modelo de stacking consigue el mejor resultado a pesar de ser el que con menos datos ha sido entrenado, por esto, la conjunción de multitud de métodos heterogéneos resulta una buena solución.

Por último, cabe destacar el hecho de que varios modelos mejoran los resultados de persistencia, lo que evidencia la aportación cuantitativa de las técnicas implementadas

tanto en la presente memoria como en la elaborada por Meneses Agudo et al. (2019).

CAPÍTULO 5

Conclusiones

Multitud de líneas de trabajo fueron abiertas en la memoria anterior desarrollada por Meneses Agudo et al. (2019). En ella se propusieron una metodología de ensembles, la aplicación de modelos VARMA como extensión de los ARIMA, la ampliación de las SVMs lineales a procesos gaussianos (GP) o a la kernel ridge regression (KRR) así como una aproximación a redes neuronales recurrentes (RNN) con arquitecturas más sofisticadas.

De este modo, el presente trabajo se construye como continuación natural de dicho trabajo, extendiendo su análisis y/o profundizando en algunos de los métodos utilizados con anterioridad. En particular, esta memoria da respuesta a la metodología de ensembles propuesta así como un ligero paso adelante en el mundo de las SVMs, añadiendo la no linealidad al núcleo del kernel. De hecho, de todos los métodos propuestos en el capítulo 3 las SVMs son las que reportan la mayor precisión en los resultados mostrados en el capítulo 4. Tanto en el caso de la predicción a corto como a medio plazo, las SVMs suponen una solución razonable al problema planteado mientras que, por su parte, la metodología de ensembles reporta un buen resultado en el primer caso pero empeora en mayor grado cuando se incrementa el horizonte de predicción. Por su parte, los métodos basados en árboles también proporcionan resultados satisfactorios y, como los anteriores, se presentan como una buena herramienta para implementar en un entorno productivo. Los GAM tienen más dificultades para competir con los métodos anteriores ya que presentan más problemas para plantear soluciones tratándose de conjuntos de datos grandes, estos problemas quedan recogidos por Wood et al. (2015), que proponen distintas metodologías para abordar su solución.

Tras esta aplicación, quedan abiertas las líneas de trabajo propuestas por Meneses Agudo et al. (2019) que se corresponden con la aplicación de procesos gaussianos (Girard et al., 2003), kernel ridge regresión (Exterkate et al., 2016) o arquitecturas más complejas de redes neuronales recurrentes (Selvin et al., 2017). Otras líneas de trabajo que sería interesante investigar en este contexto sería, por un lado, la implementación de modelos híbridos que permitiría recoger de mejor manera los patrones lineales y no

lineales de una serie temporal (Faruk, 2010). Por su parte, la exploración de diferentes configuraciones de un modelo stacking de ensembles podría solucionar varios de sus problemas de configuración derivados de su heterogeneidad. Varios estudios defienden que un modelo de ensembles heterogéneo como el presentado en esta memoria suponen una mejor solución para problemas de predicción (Tsai et al., 2011) que una configuración más homogénea, lo que sería interesante constatar.

Por último, cabe destacar que la presente memoria responde al planteamiento de un problema de carácter industrial que es necesario resolver a partir de métodos integrables de forma eficiente en un sistema de producción. Desde esta perspectiva, el modelo stacking supone un modelo candidato a su implementación en este contexto, si bien su heterogeneidad podría suponer un problema en sus niveles de adaptabilidad y escalabilidad.

Bibliografía

ABRIL, J. C. (2011). Análisis de la evolución de las técnicas de series de tiempo: Un enfoque unificado.

AGUILAR BARREIRO, P. (2019). Modelos aditivos generalizados.

BAUER, E. and KOHAVI, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. 36(1-2):105–139.

BUSSETI, E., OSBAND, I., and WONG, S. (2012). Deep learning for time series modeling. pp. 1–5.

CHEN, B.-J., CHANG, M.-W., ET AL. (2004). Load forecasting using support vector machines: A study on eunite competition 2001. 19(4):1821–1830.

CHEN, T. and GUESTRIN, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.

CHOULDECHOVA, A. and HASTIE, T. (2015). Generalized additive model selection.

CORTES, C. and VAPNIK, V. (1995). Support-vector networks. 20(3):273–297.

CRONE, S. F., GUAJARDO, J., and WEBER, R. (2006). The impact of preprocessing on support vector regression and neural networks in time series prediction. In *DMIN*, pp. 37–44.

DAMBORG, M., EL-SHARKAWI, M., AGGOUNE, M., and MARKS, R. (1990). Potential of artificial neural networks in power system operation. In *IEEE International Symposium on Circuits and Systems*, pp. 2933–2937. IEEE.

DOROGUSH, A. V., ERSHOV, V., and GULIN, A. (2018). Catboost: gradient boosting with categorical features support.

DURBÁN, M. (2008). Modelos aditivos generalizados con p-splines.

EILERS, P. H. and MARX, B. D. (1996). Flexible smoothing with b-splines and penalties. pp. 89–102.

ELRAZAZ, Z. and MAZI, A. (1989). Unified weekly peak load forecasting for fast growing power system. In *IEE Proceedings C (Generation, Transmission and Distribution)*, vol. 136, pp. 29–34. IET.

EXTERKATE, P., GROENEN, P. J., HEIJ, C., and VAN DIJK, D. (2016). Nonlinear forecasting with many predictors using kernel ridge regression. 32(3):736–753.

FARUK, D. Ö. (2010). A hybrid neural network and arima model for water quality time series prediction. 23(4):586–594.

FLETCHER, T. (2009). Support vector machines explained. p. 4.

FREUND, Y. (1995). Boosting a weak learning algorithm by majority. 121(2):256–285.

FREUND, Y., SCHAPIRE, R. E., ET AL. (1996). Experiments with a new boosting algorithm. In *icml*, vol. 96, pp. 148–156. Citeseer.

FRIEDMAN, J. H. (2001). Greedy function approximation: a gradient boosting machine. pp. 1189–1232.

GIRARD, A., RASMUSSEN, C. E., CANDELA, J. Q., and MURRAY-SMITH, R. (2003). Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. In *Advances in neural information processing systems*, pp. 545–552.

GRAF, A. B. and BORER, S. (2001). Normalization in support vector machines. In *Joint pattern recognition symposium*, pp. 277–282. Springer.

HANSEN, L. K. and SALAMON, P. (1990). Neural network ensembles. 12(10):993–1001.

HASTIE, T. and TIBSHIRANI, R. (1986). Generalized additive models. *Statistical Science*, 1(3):297–310. URL `https://doi.org/10.1214/ss/1177013604`.

HE, W. (2017). Load forecasting via deep neural networks. 122:308–314.

HUANG, S. and WU, T. (2008). Integrating ga-based time-scale feature extractions with svms for stock index forecasting. 35(4):2080–2088.

HUANG, W., NAKAMORI, Y., and WANG, S.-Y. (2005). Forecasting stock market movement direction with support vector machine. 32(10):2513–2522.

HYNDMAN, R. J. and KOEHLER, A. B. (2006). Another look at measures of forecast accuracy. 22(4):679–688.

JHAVERI, S., KHEDKAR, I., KANTHARIA, Y., and JASWAL, S. (2019). Success prediction using random forest, catboost, xgboost and adaboost for kickstarter campaigns. In *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 1170–1173. IEEE.

JUBERIAS, G., YUNTA, R., MORENO, J. G., and MENDIVIL, C. (1999). A new arima model for hourly load forecasting. In *1999 IEEE Transmission and Distribution Conference (Cat. No. 99CH36333)*, vol. 1, pp. 314–319. IEEE.

KE, G., MENG, Q., FINLEY, T., WANG, T., CHEN, W., MA, W., YE, Q., and LIU, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pp. 3146–3154.

MENESES AGUDO, C. A. ET AL. (2019). Análisis y predicción de series temporales provenientes de un sistema scada de una planta de fabricación industrial.

O'SULLIVAN, F. (1986). A statistical perspective on ill-posed inverse problems. pp. 502–518.

ROBINZONOV, N., TUTZ, G., and HOTHORN, T. (2012). Boosting techniques for nonlinear time series models. 96(1):99–122.

SELVIN, S., VINAYAKUMAR, R., GOPALAKRISHNAN, E., MENON, V. K., and SOMAN, K. (2017). Stock price prediction using lstm, rnn and cnn-sliding window model. In *2017 international conference on advances in computing, communications and informatics (icacci)*, pp. 1643–1647. IEEE.

SINGH, A. K., KHATOON, S., MUAZZAM, M., CHATURVEDI, D., ET AL. (2012). Load forecasting techniques and methodologies: A review. In *2012 2nd International Conference on Power, Control and Embedded Systems*, pp. 1–10. IEEE.

SMOLA, A. J. and SCHÖLKOPF, B. (2004). A tutorial on support vector regression. 14(3):199–222.

TSAI, C.-F., LIN, Y.-C., YEN, D. C., and CHEN, Y.-M. (2011). Predicting stock returns by classifier ensembles. 11(2):2452–2459.

WOLPERT, D. H. (1992). Stacked generalization. 5(2):241–259.

WOOD, S. N., GOUDE, Y., and SHAW, S. (2015). Generalized additive models for large data sets. 64(1):139–155.